REMARKS

Applicants add new claims 23-29. Applicants perform clarifying amendments to the claims. The amendments and new claims are supported by the specification and drawings and in particular page 8, lines 16-26, page 13, lines 4-9, the original claims, and FIGS. 2 and 4.

Applicants respectfully submit that the claims prior to amendment herein are patentable over the cited references. The amendments made herein are merely for sake of clarity and the claims should retain their full range of equivalents. Applicants provide arguments herein based on the claims prior to amendment.

CLAIMS 1, 16, 18, 21, 22

Claims 1, 18, 21, and 22 are independent claims, and each of these independent claims recites similar subject matter. Claim 1 is chosen below for the following argument as being representative. Prior to proceeding to review claim 1 and presenting the argument, an overview of exemplary embodiments will be given.

Exemplary embodiments of the disclosed invention attempt to solve the problems detailed at page 2, line 18 to page 6, line 19. In particular, Applicants have determined that for a system including mobile terminals, many sessions can be created. These sessions can remain open for days or even weeks and can be active or inactive. See page 5, lines 7-26. Furthermore, a gateway server has difficulty managing such a large number of sessions. "[A]ssigning one thread to each session is an inefficient use of system resources. On the other hand, having only one thread to handle all events of all sessions is also inefficient because the thread may not process the events more quickly than they are generated in the protocol stack." Page 6, lines 1-4. Moreover, allocating several threads to the same session also has associated problems, as detailed at page 6, lines 6-19.

The Applicants determined that grouping sessions and assigning a thread to each group of sessions would solve or at least ameliorate the aforementioned and other problems. FIG. 4 of the application is presented below:

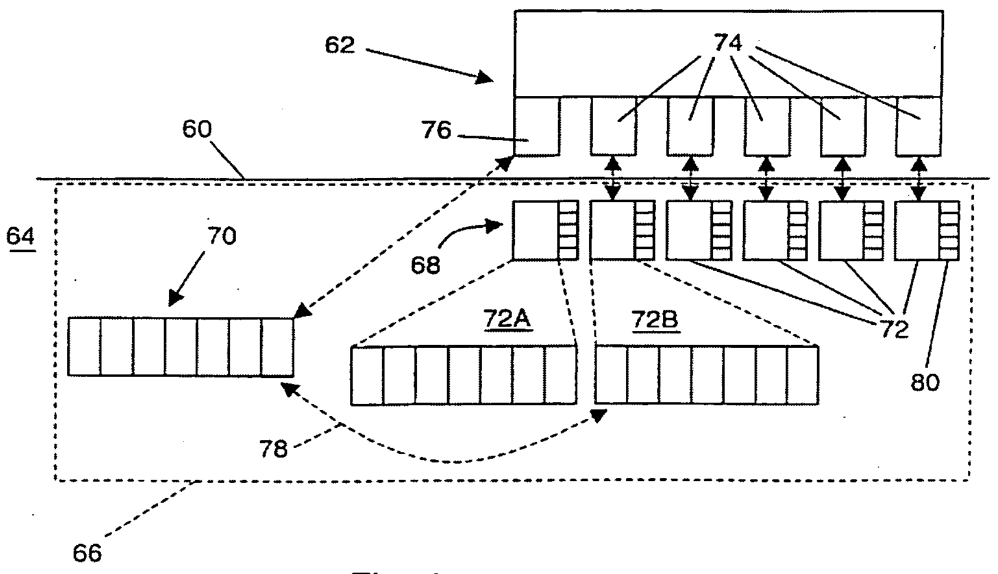


Fig. 4

In this example, unassigned sessions 70 are assigned by an acceptor thread 76 to one of the threads 74. Each of the threads 74 assigned to a group 72A, 72B of sessions. Each group of sessions has a queue 80 that contains events arising in that group.

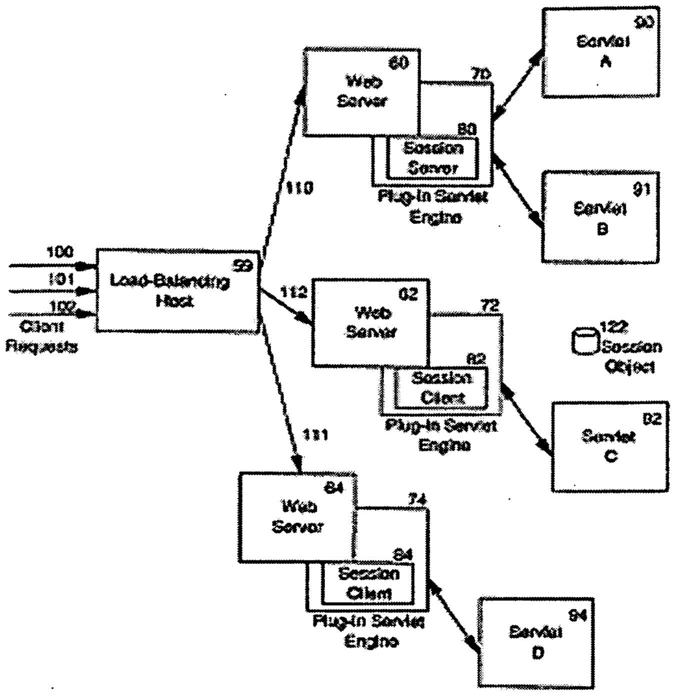
In light of this discussion, claim 1 recites the following:

> A method of managing a plurality of sessions, the sessions being between a plurality of terminals and a server having a plurality of threads, the method comprising:
>
> grouping the sessions into a plurality of groups; and
>
> assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions.

It is respectfully submitted that none of the cited references includes the unique features of independent claim 1 and in particular the subject matter of "assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions." It is noted that a thread is assigned to each group of ***sessions***, which means that each group corresponds to multiple sessions in the subject matter of "assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions".

With regard to some of the terminology used in the claims, Applicants state that "[a] ***session*** is a series of interactions between a terminal and a server having a well-defined beginning and end and involving agreed-upon characteristics." Page 2, lines 19-21 (emphasis added). Applicants also state "A ***thread*** is basically a path of execution through a program and can be the smallest unit of execution that is scheduled on a processor. A thread consists of a stack, the state of the CPU registers, and an entry in the execution list of the system scheduler." Page 4, lines 10-13 (emphasis added).

With regard to the rejections in the final Office Action, Bayeh is directed to spreading requests among a number of servlets/web servers. FIG. 3 of Bayeh is shown below.



In Bayeh, the requests 100-102 are passed through a load balancing host 59, which sends the requests to web servers 60, 62, and 64. The load balancing host 59 sends requests to a "server selected according to policies implemented in the load-balancing host software." Bayeh, col. 8, lines 42-58. It is believed that the "policies" are based on load of the web server and requests are sent to a web server based on load. The load balancing host 59 is not disclosed or implied as being one that would "group" the requests based on session. In fact, Bayeh appears to disclose that the load balancing host 59 acts only on requests and it is immaterial for purposes of balancing load as to which session it is that a request is related.

Because requests are spread among a number of servlets 90, 91, 92, and 94 such that any servlet can handle requests from any session, more than one servlet might be able to access — at the same time — session information for a particular session. Bayeh discloses techniques for ensuring that only one servlet 90-93, 94 can access session information at any time for a particular session while requests 100-102 can still be directed to any servlet. See, e.g., FIGS. 3, 4A, and 4B of Bayeh, and in particular steps 410-480.

In Bayeh therefore, there is no concerted effort or implication of grouping sessions into groups and assigning a thread to each group of sessions. Consequently, there is no disclosure or implication in Bayeh of "grouping the sessions into a plurality of groups" or "assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions" as recited in claim 1.

Freund also does not disclose or imply the recited subject matter from independent claim 1. FIG. 2 of Freund is shown below.

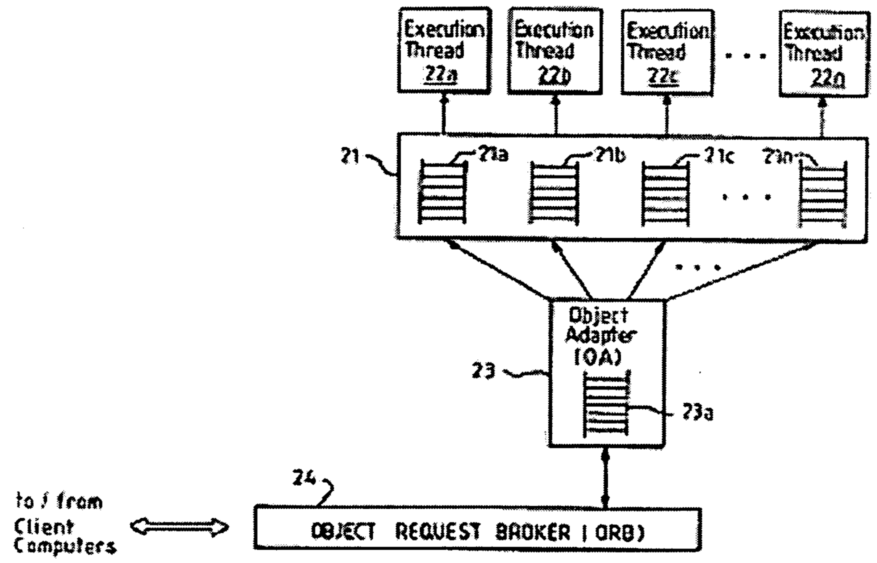


FIG. 2

What Freund appears to disclose is a system for ensuring that all related requests (e.g., related through a specific transaction) are sent to the same thread. See, e.g., the following section of Freund:

> A first embodiment of the server architecture (FIG. 2) involves the placing of a group 21 of FIFO queues 21a-21n with one request queue assigned to each execution thread 22a-22n in a one-to-one relationship. According to this embodiment, when client requests are received by the server's Object Adapter 23 over the Object Request Broker 24 from a client

> computer system, the Object Adapter 23 examines the contents of each request contained on its received request FIFO buffer 23a. Based on such contents the requests can then be forwarded on to the appropriate request queue 21a-21n. For example, if a first received client request relates to a particular transaction and a second received client request relates to a different transaction, the first request can be assigned to queue 21a (and its corresponding execution thread 22a) and the second request can be assigned to queue 21b (and its corresponding execution thread 22b). Then, if a third received transaction request relates to the same transaction as the first request, the object adapter 23 would recognize this and assign this third request to the queue 21a to be processed by execution thread 22a.
>
> In this way, ***a complete transaction*** consisting of many separate (but related) requests can be executed ***by the same execution thread***, thus providing the same execution environment for each transactionally related request.

Freund col. 5, lines 3-27 (emphasis added).

Freund describes a transaction as the following: "A transaction defines a single unit of work that must either be fully completed or fully purged without action". Freund, col. 3, lines 11-12. Freund also states the following: "According to these various embodiments, a scheduling mechanism ... ensures that all requests that are related (e.g. part of the same transaction) are sent to the same execution thread for processing." Freund, col. 6, lines 39-43. The Examiner's arguments imply that a "transaction" in Freud is equivalent to a "session" in Applicants' claims (which Applicants do not admit).

There is no teaching or implication in Freund that multiple "transactions" are assigned to a single thread. In fact, it appears in Freund that a single thread is assigned to a single transaction:

> According to these various embodiments, a scheduling mechanism (which does not necessarily have to be located in the Object Adapter) ***ensures that all requests that are related*** (e.g. part of the same transaction) ***are sent to the same execution thread for processing***. This ensures consistency during the processing of an entire set of related requests. That is, the client machine issuing a sequence of transactionally related requests of the server machine can expect to get the same answer back when it

> issues the same sequence of requests at a later time. The processing conditions of the server's execution environment will stay the same because of the scheduling mechanism. That is, ***intermediate requests belonging to another transaction (or not related to a transaction at all) are not allowed to be processed by the execution thread currently processing a transaction***. If such intermediate requests were allowed to be processed on a transaction's execution thread, the execution environment would be different when later parts of the transaction are processed by the thread and consistent results to report back to the client would not be guaranteed.
>
> In order to determine whether a request belongs to a transaction, and the specifics of the transaction if it does, the Object Request Broker (ORB) 24 interrogates the transaction context of each incoming request. The transaction context of a request is obtained by the ORB by using the OMG-established Object Transaction Service (OTS) [OMG document 94.8.4 published in 1994]. The ORB also interrogates the Object Reference and any Service Contexts of the request to determine the specific server object (and thus server application) which the request is wishing to invoke. Once the transaction context and server context/application are determined, the ORB sends the request to the appropriate Object Adapter's queue. From there, the scheduling mechanism, as described in the above embodiments, ensures that all transactionally related requests are sent to the same execution thread. Also, ***the scheduling mechanism can isolate the execution thread for a particular transaction by not allowing requests unrelated to that transaction from being processed on the transaction's assigned execution thread.***

Freund, col. 6, line 39 to col. 7, line 10 (emphases added). Consequently, Freund discloses that a single thread is assigned to a single "transaction" and there is no disclosure or implication that transactions are grouped and that a thread is assigned to a group of transactions.

Therefore even if a "transaction" in Freund is considered to be a "session" as used in the claims herein (which Applicants do not admit), Freund does not disclose or imply "grouping the sessions into a plurality of groups" or "assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions" as recited in claim 1.

Because neither Bayeh nor Freud alone discloses "grouping the sessions into a plurality of groups" or "assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions", the combination of Bayeh and Freund does not disclose this subject matter.

For at least these reasons, none of Bayeh, Freund, or their combination discloses or implies the subject matter from claim 1 of "grouping the sessions into a plurality of groups" or "assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions".

Furthermore, there does not appear to be motivation for combination of Bayeh with Freund, as Bayeh specifically allows requests from a single session to be spread amongst a number of servlets and therefore provides a mechanism for allowing multiple servlets to access session information from a single session. In other words, any servlet could be sent a request and access information from a single session corresponding to that request. By contrast, the system in Freund appears to send all requests from a single transaction to a single thread (e.g., "the scheduling mechanism, as described in the above embodiments, ensures that all transactionally related requests are sent to the same execution thread"; Freund, col. 7, lines 4-7). Therefore, the system of Freund would not need the mechanism of Bayeh, as only a single thread in Freund would handle requests from a single transaction. In Freund, multiple threads would ***not*** attempt to access session information from a single transaction, whereas allowing multiple servlets to access session information for a single session is the main idea in Bayeh.

Consequently, there is no motivation to combine Bayeh and Freund and the purported combination of Bayeh and Freund is invalid. Because the combination is invalid, even if a combination of Bayeh and Freund discloses all elements in independent claim 1 (which Applicants submit are not disclosed by a combination of Bayeh and Freund), the rejection under §103 is improper because the combination itself of Bayeh and Freund is invalid.

However, the Examiner asserts the following:

> By utilizing the queue assignment techniques of Freund to assign sessions to particular web servers, servlets and threads, the system of Bayeh is enhanced by allowing each session to be executed in the same environment as before, resulting in reduced overhead processing for the servers.

Outstanding final Office Action, page 9, section 21. Even if the above assertion were true (which Applicants do not admit), the combination of Bayeh and Freund still would not disclose the subject matter of "grouping the sessions into a plurality of groups" or "assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions", as neither reference singly discloses this subject matter.

Moreover (as stated above), if the queue assignment techniques of Freund were utilized to assign sessions to particular web servers in Bayeh, it appears this would render the invention in Bayeh superfluous or inoperable. As described above, Bayeh is directed to techniques for ensuring that only one servlet can access session information at any time for a particular session while requests can still be directed to any servlet. See, e.g., FIGS. 3, 4A, and 4B of Bayeh, and in particular steps 410-480. The main techniques in the system of Bayeh occur because requests can be directed (e.g., by the load-balancing host 59) to any servlet. By contrast, the queue assignment techniques of Freund "distribut[e] client requests stored in said buffer to said plurality of execution threads in a manner such that related client requests are sent to the same execution thread". Freund, Abstract. Therefore, in Freund, no mechanism is necessary for ensuring that only one thread can access transaction information at any time for a particular transaction because ***only*** one thread will access transaction information at any particular time for a particular transaction. The invention in Bayeh, which allows multiple servlets to access the same session information, would be rendered superfluous or inoperable by the combination of Bayeh and Freund.

For at least these reasons, the combination of Bayeh and Freund is invalid. Claims 1-22 are therefore patentable.

With regard to independent claims 18 and 22, each of these claims recites subject matter similar to the subject matter in independent claim 1. In particular, independent claim 18 recites "A server for managing a plurality of sessions with a plurality of terminals, the server comprising a plurality of threads, grouping means to group the sessions into a plurality of groups, and assigning means to assign a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions"; independent claim 21 recites "A communications system comprising a server and a plurality of terminals, the server and the terminals conducting a plurality of sessions, the server comprising a plurality of threads, grouping means to group the sessions into a plurality of groups and assigning means to assign at least one thread to each group of sessions so that the assigned thread only handles the events of that group of sessions"; and independent claim 22 recites "A computer program product for managing a plurality of sessions, the sessions being between a plurality of terminals and a server having a plurality of threads, comprising: computer readable program means for grouping the sessions into a plurality of groups; and computer readable program means for assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions".

Therefore, each of independent claims 1, 18, and 22 is patentable over the (improper) combination of Bayeh and Freund. Because independent claims 1, 18, and 22 are patentable, each of dependent claims 2-17, 19, and 20 are also patentable for at least the reasons given above.

<u>CLAIM 2</u>

Dependent claim 2 recites "A method according to claim 1 in which grouping [of sessions into a plurality of groups] occurs when a session is created."

Bayeh does not disclose or imply grouping sessions into a plurality of groups, and therefore Bayeh does not disclose that grouping occurs when a session is created. The load-balancing host 59 in Bayeh does not "group" requests when sessions are created. Instead, the load-balancing host 59 sends requests 100-102 to servers 60, 62, 64 based on load and it is immaterial as to which session the request relates. See Bayeh, col. 8, lines 49-53. In fact, a request may not be related to a session until *after* the load-balancing host 59 has sent the request to a web server in Bayeh. See Bayeh at col. 10, lines 10-13.

In Freund, again there is no grouping of sessions into a plurality of groups, and therefore there is no "grouping [of sessions into a plurality of groups] occurs when a session is created". Instead, each "transaction" in Freund is assigned to a single thread 22. See, e.g., Freund, col. 6, lines 39-43:

> According to these various embodiments, a scheduling mechanism (which does not necessarily have to be located in the Object Adapter) ***ensures that all requests that are related (e.g. part of the same transaction) are sent to the same execution thread for processing***.

Freund, col. 6, lines 39-43 (emphasis added). See also col. 7, lines 7-10 of Freund: "Also, ***the scheduling mechanism can isolate the execution thread for a particular transaction by not allowing requests unrelated to that transaction from being processed on the transaction's assigned execution thread.***" Therefore even if one could argue that a "transaction" in Freund is considered to be a "session" of the claims, the transactions are not grouped into a plurality of groups and instead a single transaction is assigned to a single thread.

For at least these reasons, dependent claim 2 is not rendered obvious by and is patentable over the (invalid) combination of Bayeh and Freund.

CLAIM 3

Dependent claim 3 recites "A method according to claim 1 in which grouping [of sessions into a plurality of groups] occurs when a session becomes active."

For the reasons given above with respect to claim 2, claim 3 is patentable over Bayeh, Freund, or their alleged combination. To reiterate, Bayeh does not disclose or imply grouping sessions into a plurality of groups, and therefore Bayeh does not disclose that grouping occurs when a session becomes active. The load-balancing host 59 in Bayeh does not "group" requests when sessions are created. Instead, the load-balancing host 59 sends requests 100-102 to servers 60, 62, 64 based on load and it is immaterial as to which session the request relates. See Bayeh, col. 8, lines 49-53. In fact, a request may not be related to a session until *after* the load-balancing host 59 has sent the request to a web server in Bayeh. See Bayeh at col. 10, lines 10-13.

In Freund, again there is no grouping of sessions into a plurality of groups, and therefore there is no "grouping [of sessions into a plurality of groups] occurs when a session becomes active". Instead, each "transaction" in Freund is assigned to a single thread 22. See, e.g., Freund, col. 6, lines 39-43:

> According to these various embodiments, a scheduling mechanism (which does not necessarily have to be located in the Object Adapter) ***ensures that all requests that are related (e.g. part of the same transaction) are sent to the same execution thread for processing.***

Freund, col. 6, lines 39-43 (emphasis added). See also col. 7, lines 7-10 of Freund: "Also, ***the scheduling mechanism can isolate the execution thread for a particular transaction by not allowing requests unrelated to that transaction from being processed on the transaction's assigned execution thread.***" Therefore even if one could argue that a "transaction" in Freund is considered to be a "session" of the claims, the transactions are not grouped into a plurality of groups and instead a single transaction is assigned to a single thread.

Furthermore, the Examiner asserts that "it is understood that when a session is created, it is inherently becoming active". However, a session can be created and become inactive then active. See, e.g., page 14, lines 23-28 of Applicants' specification: "In another embodiment of the invention, the acceptor thread assigns inactive sessions to groups when they are resumed (at that point there are no any other events for that session which need to be processed). As a result more dynamic session assignment is provided and thus more efficient session management. Inactive sessions are resumed by sending a message (or event) that the session is to be resumed." Thus, creating a session and activating a session can occur at two different times. Neither Bayeh nor Freund discloses or implies that sessions become inactive and then active again.

For at least these reasons, dependent claim 3 is not rendered obvious by and is patentable over the (invalid) combination of Bayeh and Freund.

Claim 4

Dependent claim 4 recites "A method according to claim 1 in which one group is provided for each thread so that there are equal numbers of groups and threads." The Examiner states that Bayeh discloses the subject matter in claim 4 because there are equal numbers of groups, which are web servers receiving requests, and threads, which are servlet engines which will participate in the session management solution.

In the rejection of independent claim 1, the Examiner appeared to equate a "servlet" of Bayeh with a "thread" of independent claim 1. In the rejection of claim 4, the Examiner apparently changes this such that a "servlet engine" is equated with a "thread" of independent claim 1. This is confusing as Bayeh discloses both a servlet engine and a servlet (see FIG. 3 of Bayeh, where there is a plug-in servlet engine 72 and a servlet 92) and it appears that multiple servlets can correspond to a single servlet engine (i.e., servlet engine 70 in Bayeh corresponds to servlets 90 and 91).

Regardless, neither a servlet nor a servlet engine in Bayeh is assigned to a group of sessions. As described above, requests are routed by a load-balancing host 59 in Bayeh to web servers 60, 62, 64 based on load balancing, but a request may or may not be related to a session. There is no grouping of sessions by a servlet 90-92, 94 or a servlet engine 70, 72, 74 or a load-balancing host 59 in Bayeh. In Bayeh, there does appear to be one servlet engine per web server 60, 62, 64, but that servlet engine can operate on servlets corresponding to any session. See, e.g., Bayeh at col. 10, line 4 to col. 11, line 8 and FIG. 4 and associated text.

In Freund, again there is no grouping of sessions into a plurality of groups, and therefore there is no "grouping [of sessions into a plurality of groups] occurs when a session becomes active". Instead, each "transaction" in Freund is assigned to a single thread 22. See, e.g., Freund, col. 6, lines 39-43:

> According to these various embodiments, a scheduling mechanism (which does not necessarily have to be located in the Object Adapter) ***ensures that all requests that are related (e.g. part of the same transaction) are sent to the same execution thread for processing***.

Freund, col. 6, lines 39-43 (emphasis added). See also col. 7, lines 7-10 of Freund: "Also, ***the scheduling mechanism can isolate the execution thread for a particular transaction by not allowing requests unrelated to that transaction from being processed on the transaction's assigned execution thread.***" Therefore even if one could argue that a "transaction" in Freund is considered to be a "session" of the claims, the transactions are not grouped into a plurality of groups and instead a single transaction is assigned to a single thread.

For at least these reasons, dependent claim 4 is not rendered obvious by and is patentable over the (invalid) combination of Bayeh and Freund.

CLAIM 5

Prior to proceeding to the rejections regarding claim 5, it is helpful to review some amount of prosecution history. The Examiner issued a non-final Office Action on 7 April 2006. In this non-final Office Action, the Examiner issued a number of Official Notice statements related to the dependent claims. The Applicants responded in a Response dated 5 July 2006, in which the arguments related to the independent claims were considered to be fully dispositive of the issues and therefore no comments regarding the dependent claims where made.

However, in the final Office Action dated 9 August 2006, the Examiner stated the following:

> Applicant has failed to seasonably challenge the Examiner's assertions of well known subject matter in the previous Office action(s) pursuant to the requirements set forth under MPEP §2144.03. A "seasonable challenge" is an explicit demand for evidence set forth by Applicant in the next response. Accordingly, the claim limitations the Examiner considered as "well known" in the first Office action are now established as admitted prior art of record for the course of the prosecution. See In re Chevenard, 139 F.2d 71, 60 USPQ 239 (CCPA 1943).

Applicants respectfully disagree for at least the following reasons.

The Examiner asserts that "the claim limitations the Examiner considered as 'well known' in the first Office action are now established as admitted prior art of record ***for the course of the prosecution***". M.P.E.P. §2144.03 states the following:

> If applicant does not traverse the examiner's assertion of official notice or applicant's traverse is not adequate, the examiner should clearly indicate in the next Office action that the common knowledge or well-known in the art statement is taken to be admitted prior art because applicant either failed to traverse the examiner's assertion of official notice or that the traverse was inadequate. If the traverse was inadequate, the examiner should include an explanation as to why it was inadequate.

M.P.E.P. §2144.03 (Rev. 3, Aug. 2005). Although the M.P.E.P. indicates that the "well-known in the art statement is taken to be admitted prior art", the M.P.E.P. does not have the force of law or rule: "The Manual ***does not have the force of law*** or the force of the rules in Title 37 of the Code of Federal Regulations." M.P.E.P., the section entitled "Forward" (Rev. 3, Aug. 2005). Applicants can find no case law, statute, or rule indicating that an initial failure to respond to a statement of Official Notice in an Office Action means that the statement of Official Notice is valid for time immemorial (i.e., the course of the prosecution, as asserted by the Examiner). This is especially true for Official Notice statements in rejections of dependent claims when responsive arguments presented as to the independent claims were dispositive of all issues. Consequently, Applicants will challenge each of these Official Notice statements as described below.

In particular, the Examiner uses terminology similar to the following (where "*A*" is some concept): "By this rationale, 'Official Notice' is taken that ***both the concept(s) and advantages*** of providing for *A* are well known" or "By this rationale, 'Official Notice' is taken that ***both the concept(s) and advantages of providing for*** *A* are well known and expected in the art." For instance, the Examiner asserts the following: "By this rationale, 'Official Notice' that both the ***concepts and advantages of providing for*** sessions which remain open until closed is well known in the art" (page 7 of the final Office Action) (emphasis added); or "By this rationale, 'Official Notice' is taken that both the ***concept and advantages of providing for*** static load balancing techniques are well known and expected in the art" (page 4 of the final Office Action) (emphasis added).

Applicants respectfully submit that this terminology is unclear, as the metes and bounds of these statements simply cannot be determined. As an example, consider the statement of "By this rationale, 'Official Notice' is taken that it is well known that *the sky is blue*", where the *sky is blue* is the asserted concept "*A*". This statement is clear, as Applicants can determine whether the sky is or is not blue. Now consider the statement of "By this rationale, 'Official Notice' is taken that both the concept(s) and advantages of providing for *a blue sky* are well known". With the latter statement, it is unclear as to what concepts and advantages are being relied upon in the statement that "the concepts and advantages of providing for *a blue sky* are well known".

Furthermore, the phrase "concept(s) and advantages" appears to modify "providing" and not "*a blue sky*". In other words, the statement could be "By this rationale, 'Official Notice' is taken that both the concept[s] and advantages of *a blue sky* are well known"; instead, the statement is "By this rationale, 'Official Notice' is taken that both the concept(s) and advantages of <u>providing for</u> *a blue sky* are well known". It is unclear as to what the phrase "providing for" means in this context and how this phrase relates to the asserted concept (in this example, *a blue sky*) and therefore to some asserted fact. For instance, is the asserted fact that one is <u>providing for</u> a blue sky or is the asserted fact the blue sky itself (or both providing for a blue sky and the blue sky itself)?

Turning to an exemplary statement of Official Notice made by the Examiner, the Examiner asserts the following: "By this rationale, 'Official Notice' is taken that both the ***concept and advantages of providing for*** static load balancing techniques are well known and expected in the art" (page 4 of the final Office Action, in a rejection of claim 5) (emphasis added). Applicants cannot determine the metes and bounds of this statement. What are the concept and advantages being relied upon? Further, what is asserted to be well known: the <u>providing for</u> static load balancing techniques, or the static load balancing techniques themselves (or both)?

For at least these reasons, Applicants contest each one of the following assertions (made both in the non-final Office Action dated 7 April 2006 and in the final Office Action dated 9 August 2006, although the citations below are only from the final Office Action dated 9 August 2006):

> "By this rationale, 'Official Notice' is taken that both the concept and advantages of providing for static load balancing techniques are well known and expected in the art" (page 4 of the final Office Action, in a rejection corresponding to claim 5).
>
> "By this rationale, 'Official Notice' is taken that both the concept and advantages of providing for relative load balancing techniques are well known and expected in the art" (page 5 of the final Office Action, in a rejection corresponding to claim 7).
>
> "By this rationale, 'Official Notice' is taken that both the concept and advantages of providing for random load balancing techniques are well known and expected in the art" (page 6 of the final Office Action, in a rejection corresponding to claim 8).
>
> "By this rationale, 'Official Notice' that both the concepts and advantages of providing for sessions which remain open until closed is well

known in the art" (page 7 of the final Office Action, in a rejection corresponding to claim 12).

"By this rationale, 'Official Notice' that both the concepts and advantages of providing for cellular telephones and mobile terminals as the terminals is well known and expected in the art" (page 7 of the final Office Action, in a rejection corresponding to claims 13 and 14).

"By this rationale, 'Official Notice' is taken that both the concept and advantages of providing for using the WSP protocol for sessions is well known and expected in the art" (page 8 of the final Office Action, in a rejection corresponding to claim 17).

Furthermore, the Examiner asserts that "***the claim limitations*** the Examiner considered as 'well known' in the first Office action are now established as admitted prior art of record". Assuming, *arguendo*, that there is material that is now established as admitted prior art of record (which Applicants do not admit), Applicants respectfully submit that this material is limited to the Examiner's factual statements of Official Notice and does not necessarily affect the claim limitations. For instance, even if the Examiner's asserted Official Notice statements are taken as being true (which Applicants do not admit), the dependent claims are still patentable in the context of the subject matter of independent claims along with the subject matter of the dependent claims, as is shown below.

Regarding claim 5, this claim recites the following: "A method according to claim 1 in which sessions are assigned statically to particular threads." Assume for sake of argument that the following statement is true (which Applicants do not admit): "By this rationale, 'Official Notice' is taken that ... static load balancing techniques are well known and expected in the art" (page 4 of the final Office Action). This statement is used in a rejection of claim 5. While load balancing could be used with the disclosed invention, there is no mention of load balancing in claim 5. In other words, claim 5 does not recite that sessions are assigned statically to particular threads ***in order to balance load between the particular threads***, which is what the Examiner appears to argue. Therefore, claim 5 is patentable even if "load balancing techniques are well known and expected in the art".

Regarding claim 5 in conjunction with claim 1, claim 1 recites in part "grouping the sessions into a plurality of groups" and "assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions". Even if the statement of "By this rationale, 'Official Notice' is taken that ... static load balancing techniques are well known and expected in the art" is taken as true (which Applicants do not admit), it is not known or expected to statically assign sessions to particular threads, where a thread is assigned to each group of sessions so that the assigned thread only handles the events of that group of sessions. Certainly Bayeh does not disclose or imply as such, as Bayeh performs load balancing of requests to web servers based on "policies implemented in load-balancing host software" (see col. 8, lines 49-58), and there is no disclosure or implication of static assignment of sessions to particular threads, where a thread is assigned to each group of sessions so that the assigned thread only handles the events of that group of sessions.

As for Freund, this reference also does not disclose or imply that a thread handles a group of sessions and therefore does not disclose or imply that sessions are assigned statically to particular threads (where each thread handles a group of sessions).

The Examiner asserts that one could modify Bayeh to include static load balancing techniques, but even if one could modify Bayeh to include static load balancing techniques, those techniques would be related to static load balancing of requests to web servers and not related to static assignment of sessions to particular threads, where a thread is assigned to each group of sessions so that the assigned thread only handles the events of that group of sessions, as recited generally in claims 5 and 1.

Therefore, claim 5 is patentable over Bayeh, Freund, and the Examiner's Official Notice.

<u>CLAIM 6</u>

Dependent claim 6 recites "A method according to claim 1 in which a session is put into a first group in a first time period before suspension and put into a second group in a second time period following resumption." The Examiner appears to be using hindsight analysis in the rejection to claim 6. Not only is there no disclosure or implication in Bayeh or Freund that a session can be suspended and then subsequently resumed, there is no disclosure or implication in Bayeh or Freund that upon resumption a session would be placed into a second group (and therefore assigned to a different thread). For instance, Bayeh sends requests to servers based on what appears to be solely load balancing techniques and there is no disclosure that sessions related to the requests are grouped in any way, and Freund appears to send requests related to a transaction always to the same thread (see, e.g., Freund, col. 6, line 39 to 43 and col. 7, lines 4-10). There is simply no disclosure or indication in Freund that transactions would be given to a different thread than the thread originally assigned to the transaction. Moreover, Applicants state the following:

> Preferably, the invention relates to long-lived sessions, in some cases some of the sessions are alive for as long as months.
>
> . . .
>
> In an alternative embodiment a session can be put into a first group in a first time period before suspension and put into a second group in a second time period following resumption. The second group can be chosen randomly or on the basis of pre-determined selection criteria, for example based upon the relative levels of activity of the groups.

Applicants' specification at page 7, line 30 to page 8, line 10. Without hindsight analysis, as neither Bayeh nor Freund gives any indication that sessions can be suspended and subsequently resumed, it should not be considered obvious that a session can be suspended and subsequently resumed, unless the Examiner uses the Applicants' specification. This is hindsight analysis. Furthermore, even if threads could be suspended and subsequently resumed, it should not be considered obvious that a session would be placed into a different thread after resumption. There is certainly no teaching or implication in either Bayeh or Freund of this.

Because the Examiner admits that Bayeh does not disclose the subject matter of claim 6, Freund does not disclose or imply the subject matter of claim 6, and the Examiner appears to be using hindsight analysis, claim 6 is not rendered obvious by, and is patentable over, Bayeh, Freund, or their alleged combination.

Claim 7

Regarding claim 7, this claim recites "A method according to claim 6 in which the second group is chosen on the basis of the relative levels of activity of the groups". Claim 6 recites "A method according to claim 1 in which a session is put into a first group in a first time period before suspension and put into a second group in a second time period following resumption". Claim 1 recites in part "grouping the sessions into a plurality of groups" and "assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions".

Claim 7 is a claim where the Examiner gave a statement of Official Notice of the following:

> By this rationale, 'Official Notice' is taken that both the concept and advantages of providing for relative load balancing techniques are well known and expected in the art

Page 5 of the final Office Action.

Even if the statement of "By this rationale, 'Official Notice' is taken that ... relative load balancing techniques are well known and expected in the art" is true (which Applicants do not admit), claim 7 is still patentable, as it is not known or expected to choose a second group on the basis of the relative levels of activity of the groups, where a session is put into a first group in a first time period before suspension and put into a second group in a second time period following resumption, and where a thread is assigned to each group of sessions so that the assigned thread only handles the events of that group of sessions The Examiner asserts that one could modify Bayeh to include [relative] load balancing techniques (note that the Examiner only says that one could modify Bayeh to include "load balancing techniques", but it appears that the term "relative load balancing techniques" is implied).

Even if one could modify Bayeh to include relative load balancing techniques, those techniques would be related to relative load balancing of requests to web servers and not to choosing a second group on the basis of the relative levels of activity of the groups, where a session is put into a first group in a first time period before suspension and put into a second group in a second time period following resumption, and where a thread is assigned to each group of sessions so that the assigned thread only handles the events of that group of sessions, as recited generally in claims 7, 6, and 1.

As discussed above in the rejection of claim 6, neither Bayeh nor Freund gives any indication that sessions can be suspended and subsequently resumed, and it should not be considered obvious that a session can be suspended and subsequently resumed or what happens when a session is resumed.

Moreover, even if load balancing techniques are well known to those skilled in the art, the load balancing described in Bayeh relates to load balancing of *requests*, which may or may not relate to sessions when the requests are load-balanced. In other words, Bayeh does not appear to perform load balancing of *sessions*. Further, Applicants recite in independent claim 1 grouping *sessions* into a plurality of groups. As described above in reference to claim 6, Bayeh does not disclose or imply that sessions can be suspended and subsequently resumed. In claim 7, a session is placed into a second group following resumption and this second group is chosen based on relative activity levels of the groups of sessions. Again, claim 7 relates to sessions and groups of sessions and Bayeh relates to load balancing of *requests*, which may or may not be related to sessions when the balancing occurs.

Although Freund does state that "[a] large number of users (clients) can thus be given efficient usage of the available resources through system-wide workload balancing" (Freund, col. 7, lines 14-17), Freund appears to send requests related to a transaction always to the same thread (see, e.g., Freund, col. 6, line 39 to 43 and col. 7, lines 4-10). There is simply no disclosure or indication in Freund that transactions would be assigned to a different thread than the thread originally assigned to the transaction. Meanwhile, claim 7 would have sessions that are resumed placed into different groups and therefore assigned to different threads.

For at least these reasons, claim 7 is patentable over Bayeh, Freund, and the Examiner's Official Notice.

CLAIM 8

Regarding claim 8, this claim recites "A method according to claim 6 in which the second group is chosen randomly". Claim 6 recites "A method according to claim 1 in which a session is put into a first group in a first time period before suspension and put into a second group in a second time period following resumption". Claim 1 recites in part "grouping the sessions into a plurality of groups" and "assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions".

Claim 8 is a claim where the Examiner gave a statement of Official Notice of the following:

> By this rationale, 'Official Notice' is taken that both the concept and advantages of providing for random load balancing techniques are well known and expected in the art.

Page 6 of the final Office Action.

Even if the statement of "By this rationale, 'Official Notice' is taken that ... random load balancing techniques are well known and expected in the art" is true (which Applicants do not admit), claim 8 is still patentable, it is not known or expected to choose a second group randomly, where a session is put into a first group in a first time period before suspension and put into a second group in a second time period following resumption, and where a thread is assigned to each group of sessions so that the assigned thread only handles the events of that group of sessions. The Examiner asserts that one could modify Bayeh to include random load balancing techniques. Even if one could modify Bayeh to include random load balancing techniques, those techniques would be related to random load balancing of requests to web servers and not to choosing a second group on the basis of the relative levels of activity of the groups, where a session is put into a first group in a first time period before suspension and put into a second group in a second time period following resumption, and where a thread is assigned to each group of sessions so that the assigned thread only handles the events of that group of sessions, as recited generally in claims 8, 6, and 1.

As discussed above in the rejection of claim 6, neither Bayeh nor Freund gives any indication that sessions can be suspended and subsequently resumed, and it should not be considered obvious that a session can be suspended and subsequently resumed or what happens when a session is resumed.

Moreover, even if load balancing techniques are well known to those skilled in the art, the load balancing described in Bayeh relates to load balancing of *requests*, which may or may not relate to sessions when the requests are load-balanced. In other words, Bayeh does not appear to perform load balancing of *sessions*. Further, Applicants recite in independent claim 1 grouping *sessions* into a plurality of groups. As described above in reference to claim 6, Bayeh does not disclose or imply that sessions can be suspended or resumed. In claim 7, a session is placed into a second group following resumption and this second group is chosen based on relative activity levels of the groups of sessions. Again, claim 7 relates to sessions and groups of sessions and Bayeh relates to load balancing of *requests*, which may or may not relate to sessions when the balancing occurs.

Although Freund does state that "[a] large number of users (clients) can thus be given efficient usage of the available resources through system-wide workload balancing" (Freund, col. 7, lines 14-17), Freund appears to send requests related to a transaction always to the same thread (see, e.g., Freund, col. 6, line 39 to 43 and col. 7, lines 4-10). There is simply no disclosure or indication in Freund that transactions would be given to a different thread than the thread originally assigned to the transaction. Meanwhile, claim 7 would have sessions that are resumed placed into different groups and therefore assigned to different threads.

For at least these reasons, claim 8 is patentable over Bayeh, Freund, and the Examiner's Official Notice.

<u>CLAIM 9</u>

Dependent claim 9 recites "A method according to claim 1 in which each group has a queue and each session puts its events into that queue." This claim indicates that a group of sessions has a queue and each session in the group puts its events into the queue. For purported disclosure of the subject matter in claim 9, the Examiner points to col. 12, lines 29-58 of Bayeh. What the cited section of Bayeh appears to disclose is techniques for ensuring that only one servlet at a time is given access to a session object. For instance, Bayeh states:

> The second tier is a First-In, First-out (FIFO) queue for each session object, where the queue is used to keep track of servlet threads that are waiting for access to this session object. In the single-tiered approach mentioned previously, the FIFO queues would suffice to implement the locking function. ***In this approach, the FIFO queue would contain a first entry for the servlet process currently holding the lock, as well as subsequent entries for any waiting processes.*** Then, the locked or unlocked status for a particular queue would be determined by checking to see if the FIFO wait queue was empty (empty indicating that the object is unlocked). Mechanisms for implementing FIFO queues and locking are well known to one of ordinary skill in the art, and will not be explained in detail.

Bayeh at col. 12, lines 44-58 (emphasis added). The cited text from Bayeh does not disclose or imply that "each group (72) [of sessions] has a queue (80) and each session puts its events into that queue (80)" as recited in dependent claim 9.

Freund does disclose the following:

> A first embodiment of the server architecture (FIG. 2) involves the placing of a group 21 of ***FIFO queues 21a-21n with one request queue assigned to each execution thread 22a-22n in a one-to-one relationship***. According to this embodiment, when client requests are received by the server's Object Adapter 23 over the Object Request Broker 24 from a client computer system, the Object Adapter 23 examines the contents of each request contained on its received request FIFO buffer 23a. ***Based on such contents the requests can then be forwarded on to the appropriate request queue 21a-21n.*** For example, if a first received client request relates to a particular transaction and a second received client request relates to a different transaction, the first request can be assigned to queue 21a (and its corresponding execution thread 22a) and the second request can be assigned to queue 21b (and its corresponding execution thread 22b). Then, if a third received transaction request relates to the same transaction as the first request, the object adapter 23 would recognize this and assign this third request to the queue 21a to be processed by execution thread 22a.
>
> In this way, ***a complete transaction consisting of many separate (but related) requests can be executed by <u>the same execution thread</u>***, thus providing the same execution environment for each transactionally related request.

Freund, col. 5, lines 3-27 (emphasis added). However, each thread 22 in Freund is assigned to a ***single*** transaction and not a group of transactions. By contrast, in claim 9 a thread is assigned to a ***group*** of sessions.

For at least these reasons, dependent claim 9 is therefore patentable over Bayeh, Freund, or the alleged combination of Bayeh and Freund.

<u>CLAIM 10</u>

Claim 10 recites "A method according to claim 1 in which the sessions are grouped by a thread referred to as an acceptor thread." As described above, Bayeh does not disclose or imply that sessions are grouped into a plurality of groups, and therefore Bayeh does not disclose that "sessions are grouped by a thread referred to as an acceptor thread" as recited in dependent claim 10. For instance, discussion about threads in Bayeh centers on "servlet threads" (see, e.g., Bayeh at col. 12, line 29 to col. 15, line 6). This discussion is mainly related to how servlet threads can gain access to a session object. There is no disclosure of how a single thread (an acceptor thread) groups sessions.

In Freund, as described above, there is also no disclosure or implication of how a single thread (an acceptor thread) groups sessions, as there is no disclosure or implications of grouping of sessions. In fact, it appears that a single transaction is assigned to a single thread: See also col. 7, lines 7-10 of Freund: "Also, ***the scheduling mechanism can isolate the execution thread for a particular transaction by not allowing requests unrelated to that transaction from being processed on the transaction's assigned execution thread.***"

Claim 10, for at least these reasons, is therefore patentable over Bayeh, Freund, or the alleged combination of Bayeh and Freund.

CLAIM 11

Claim 11 recites "A method according to claim 10 in which the acceptor thread calls a function which is answered by notification that a new session has been created and then assigns the new session to a particular session group." As described above, Bayeh does not disclose or imply that sessions are grouped into a plurality of groups, and therefore Bayeh does not disclose that a new session is assigned to a particular session group as generally recited in claim 11. Further, the portion of Bayeh cited by the Examiner appears to be unrelated to calling a function and does not disclose that a function is called. Instead, the cited portion of Bayeh describes routing requests to servers:

> FIG. 3 illustrates a model of the clustered server environment in which the present invention may be practiced, and shows how this invention interacts with other components of the environment. A Web server 60 may be connected to any number of other Web servers, shown here as 62 and 64. Clustering multiple servers in this way provides for increased capacity with which HTTP requests at a Web site can be processed. A load-balancing host 59 functions as a type of front-end processor to these servers, receiving client requests 100, 101, 102 and then routing those requests (shown here as 110, 111, 112) to a server selected according to policies implemented in the load-balancing host software. Note that the requests 110, 111, 112 are shown being sent to specific Web servers: this is merely an example of a possible outcome

> of the load balancing process. Load-balancing techniques are known in the art, and do not form part of the present invention.

Bayeh, col. 8, lines 42-58.

In Freund, as described above, there is also no disclosure or implication of how a single thread (an acceptor thread) groups sessions, as there is no disclosure or implications of grouping of sessions. In fact, it appears that a single transaction is assigned to a single thread: See also col. 7, lines 7-10 of Freund: "Also, ***the scheduling mechanism can isolate the execution thread for a particular transaction by not allowing requests unrelated to that transaction from being processed on the transaction's assigned execution thread.***"

Dependent claim 11 is therefore, for at least these reasons, patentable over Bayeh, Freund, or their alleged combination.

<u>CLAIM 12</u>

Regarding claim 12, this claim recites "A method according to claim 1 in which the sessions remain open for an undetermined period of time until closed". Claim 1 recites in part "grouping the sessions into a plurality of groups" and "assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions". It is noted that originally filed claim 12 referred to a "long-lived" session.

In the final Office Action, the Examiner made the following statement:

> By this rationale, 'Official Notice' that both the concepts and advantages of providing for sessions which remain open until closed is well known in the art.

Page 7 of the final Office Action.

Even if the statement "By this rationale, 'Official Notice' that ... sessions which remain open until closed is well known in the art" (page 7 of the final Office Action, in a rejection corresponding to claim 12) is true (which Applicants do not admit), Applicants respectfully submit that the subject matter of claim 12 in conjunction with the subject matter of claim 1 is patentable. Applicants state the following:

> In a communication system comprising a gateway server and a plurality of mobile terminals, establishing a session requires a relatively large amount of bandwidth because a terminal and a server must negotiate many characteristics relevant to the session. Furthermore, information which is unique to a particular opened session may be lost if the session is terminated. This unique information could have been negotiated as a result of transactions. For example, it may be the status of a game. In order to avoid opening and closing sessions on demand and establishing new sessions whenever they are needed, the sessions may be kept open for a long time, even in an inactive state, so that they can be resumed when needed. A session can remain open for days or even weeks until it is closed or until the terminal no longer receives power, for example from a battery. The state of a session can be preserved and kept alive even after switching the terminal off.
>
> In this case, the session state can be saved to persistent memory or a SIM card before turning the power off. Of course, although the session is still alive, is not active afterwards. As a consequence, a gateway server serving a large number of mobile terminals needs to be able to manage a very large number of sessions indeed. The gateway may serve tens of thousands of mobile terminals or even in the region of a million mobile terminals. Generally a mobile will have one session open at one time (although there may be more), and so there can be in the region of one million sessions open at one time on the server.
>
> An application in the server will use the operating system thread management service and create a number of threads to manage these sessions. However, a gateway server has difficulty dealing with such a large number of sessions. The number of event sources is much larger than the number of threads. Since most of the sessions are inactive, only a fraction of them have events at any particular time. Therefore assigning one thread to each session is an inefficient use of system resources. On the other hand, having only one thread to handle all events of all sessions is also inefficient because the thread may not process the events more quickly than they are generated in the protocol stack.

Applicants' Application, page 5, line 7 to page 6, line 19.

Therefore, it can be seen that Applicants determined that terminals (such as mobile terminals) that create sessions that are kept open for a long, undetermined period of time (e.g., the sessions are long-lived) create problems. For instance, a typical session between a client such as a personal computer and a server will be kept open for a short, determinable time period (e.g., if the client fails to respond within a certain time, the session can timeout and be closed; the sessions are, e.g., short-lived). Meanwhile, it was the Applicants who determined the following: "The number of event sources is much larger than the number of threads. Since most of the sessions are inactive, only a fraction of them have events at any particular time. Therefore assigning one thread to each session is an inefficient use of system resources. On the other hand, having only one thread to handle all events of all sessions is also inefficient because the thread may not process the events more quickly than they are generated in the protocol stack." Id.

Applicants also state the following:

> According to a first aspect of the invention there is provided a method of managing a plurality of sessions the sessions being between a plurality of terminals and a server having a plurality of threads, the method comprising: grouping the sessions into a plurality of groups; and assigning a thread to each group of sessions.
>
> The invention is able to optimise the load of the system handling the communication by reducing the number of threads needed to process the various sessions. It can also enable spreading the load of sessions across the threads. As a result, a huge numbers of sessions can be dealt with.

Applicant's Application, page 6, lines 21-30.

Therefore, even if sessions which remain open until closed is well known in the art, claim 12 is directed to sessions that remain open for an *undetermined* period of time until closed. Further, it is Applicants who determined that assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions could provide a benefit in the case of sessions that remain open for an undetermined period of time (e.g., such that the session is long-lived).

None of the cited references discuss sessions that can remain open for indeterminate time periods. In fact, the cited references appear to teach away from sessions that can remain open for indeterminate time periods. For instance, Bayeh states the following:

> There are a number of such session properties defined for use in the Session Tracking feature of the Java Servlet API. These properties include such things as ***the length of time for which a session can be inactive before it should be considered expired, <u>and deleted</u>*** from the system; how often to check for ***inactive sessions***; whether cookies or URL rewriting is being used to implement session services; etc.

Bayeh, col. 11, lines 49-58 (emphasis added). This cited text indicates that sessions can only exist for a certain time period until they are ***deleted***.

Although Freund does not appear to refer explicitly to short-lived transactions, Freund certainly implies that the transactions would be short-lived:

> Computer implemented transaction processing systems are used for critical business tasks in a number of industries. ***A transaction defines a single unit of work that must either be fully completed or fully purged without action.*** For example, in the case of a bank automated teller machine from which a customer seeks to withdraw money, the actions of issuing the money, reducing the balance of money on hand in the machine and reducing the customer's bank balance must all occur or none of them must occur. Failure of one of the subordinate actions would lead to inconsistency between the records and the actual occurrences.

Freund, col. 3, lines 8-18 (emphasis added). See also: "Also, individual users are provided with consistent results in terms of a guaranteed response ***and guaranteed processing time*** each time a client invokes a server application located on a heterogeneous platform." Freund, col. 7, lines 16-20 (emphasis added).

Therefore dependent claim 12 (in combination with claim 1) is patentable over the cited references and the Examiner's Official Notice.

Claims 13 and 14

With regard to claims 13 and 14, claim 13 recites "A method according to claim 1 in which the terminals comprise mobile terminals", and claim 14 recites "A method according to claim 13 in which the terminals comprise cellular telephones". The Examiner stated the following:

> By this rationale, 'Official Notice' that both the concepts and advantages of providing for cellular telephones and mobile terminals as the terminals is well known and expected in the art.

Page 7 of the final Office Action.

Even if the statement of "By this rationale, 'Official Notice' that ... cellular telephones and mobile terminals [used as] the terminals is well known and expected in the art" is true (which Applicants do not admit), claims 13 and 14 are nonetheless patentable. It is Applicants who discovered the unique problems associated with mobile terminals and the sessions the terminals create:

> In a communication system comprising a gateway server and a plurality of mobile terminals, establishing a session requires a relatively large amount of bandwidth because a terminal and a server must negotiate many characteristics relevant to the session. Furthermore, information which is unique to a particular opened session may be lost if the session is terminated. This unique information could have been negotiated as a result of transactions. For example, it may be the status of a game. In order to avoid opening and closing sessions on demand and establishing new sessions whenever they are needed, the sessions may be kept open for a long time, even in an inactive state, so that they can be resumed when needed. A session can remain open for days or even weeks until it is closed or until the terminal no longer receives power, for example from a battery. The state of a session can be preserved and kept alive even after switching the terminal off.
>
> In this case, the session state can be saved to persistent memory or a SIM card before turning the power off. Of course, although the session is still alive, is not active afterwards. As a consequence, a gateway server serving a large number of mobile terminals needs to be able to manage a very large number of sessions indeed. The gateway may serve tens of thousands of mobile terminals or even in the region of a million mobile terminals. Generally

> a mobile will have one session open at one time (although there may be more), and so there can be in the region of one million sessions open at one time on the server.
>
> An application in the server will use the operating system thread management service and create a number of threads to manage these sessions. However, a gateway server has difficulty dealing with such a large number of sessions. The number of event sources is much larger than the number of threads. Since most of the sessions are inactive, only a fraction of them have events at any particular time. Therefore assigning one thread to each session is an inefficient use of system resources. On the other hand, having only one thread to handle all events of all sessions is also inefficient because the thread may not process the events more quickly than they are generated in the protocol stack.
>
> If several threads are allocated to the same session, then there is a risk that two events of the same session can be dealt with by the same thread at the same time or that the same session can be designated to two different threads at the same time. If two events of the same session are processed concurrently by different threads, this may result in inconsistencies. For example, resumption of a session might be processed faster than its suspension (even though the suspension instruction arrived before the resumption instruction). Alternatively, there may be a code fraction which handles the suspension of a session and a code fraction which handles the resumption of that session. These fractions are in the server application. If both code fractions are able to modify the same data area concurrently so that one thread runs the suspension one thread runs the resumption, this could cause inconsistent data structures since the data would be manipulated by both code fractions at the same time. It is difficult to provide programs which are able to deal with such inconsistencies.

Applicants' specification, page 5, line 7 to page 6, line 19.

Therefore, it is the Applicants who determined the unique problems regarding sessions associated with mobile terminals and cellular phones (e.g., long-lived sessions, e.g., that might remain open for an undetermined period of time), and it is Applicants who therefore determined that assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions (as recited in claim 1) is important and is not implied by Bayeh, Freund, or their asserted combination. Therefore, claims 13 and

14 are, in combination with the subject matter of independent claim 1, patentable over the combination of the cited art and the Examiner's Official Notice.

CLAIM 15

Claim 15 recites "A method according to claim 1 in which load balancing means is included in the assignment mechanism of the session." In independent claim 1, a thread is assigned to each group of sessions so that the assigned thread only handles the events of that group of sessions. The assignment mechanism therefore assigns a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions and also includes a load balancing means. Bayeh does not disclose, as described above, any assignment of a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions. The cited portion of Bayeh, which the Examiner asserts discloses the subject matter of claim 15, discloses load balancing based on requests, and a request may or may not be related to a session at the time the load-balancing occurs in Bayeh.

With regard to Freund, Freund does state that "[a] large number of users (clients) can thus be given efficient usage of the available resources through system-wide workload balancing" (Freund, col. 7, lines 14-17). However, Freund appears to send requests related to a transaction always to the same thread (see, e.g., Freund, col. 6, line 39 to 43 and col. 7, lines 4-10). There is simply no disclosure or indication in Freund that a thread handles groups of transactions and further that a load balancing means is used when assigning thread to groups of transactions.

Therefore, dependent claim 15 is, for at least these reasons, patentable over Bayeh, Freund, or their purported combination.

CLAIM 17

With regard to claim 17, this claim recites "in which the sessions are part of the Wireless Session Protocol (WSP)". The Examiner states the following in the rejection of claim 17:

> By this rationale, 'Official Notice' is taken that both the concept and advantages of providing for using the WSP protocol for sessions is well known and expected in the art.

Page 8 of the final Office Action.

Even if the statement "By this rationale, 'Official Notice' is taken that ... using the WSP protocol for sessions is well known and expected in the art" is true (which Applicants do not admit), dependent claim 17 is patentable. As described above, it is Applicants who determined problems regarding sessions using the WSP protocol (e.g., long-lived sessions, e.g., that might remain open for an undetermined period of time), and it is Applicants who therefore determined that assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions (as recited in claim 1) is important. Therefore, claim 17 is, in combination with the subject matter of independent claim 1, patentable over the alleged combination of the cited art.

CLAIMS 19 AND 20

Claim 19 recites "A server according to claim 18 comprising a gateway server serving a plurality of mobile terminals" and claim 20 recites "A server according to claim 19 comprising a WAP-HTTP gateway." It is Applicants who discovered the unique problems associated with mobile terminals and the sessions the terminals create. See Applicants' specification, page 5, line 7 to page 6, line 19. There is no mention of these problems in Bayeh, Freund, or their alleged combination, nor is there any mention of the solutions presented herein.

It is Applicants who therefore determined that assigning a thread to each group of sessions so that the assigned thread only handles the events of that group of sessions (as recited in claim 18) is important in the context of mobile terminals and WAP-HTTP gateways. Therefore, claims 19 and 20 are, in combination with the subject matter of independent claim 18, patentable over the alleged combination of the cited references.

NEW CLAIMS

It is respectfully submitted that the new claims 22-29 are patentable over the cited references for at least the reasons given above.

CONCLUSION

Based on the foregoing arguments, it should be apparent that claims 1-29 are thus allowable over the reference(s) cited by the Examiner, and the Examiner is respectfully requested to reconsider and remove the rejections. The Examiner is invited to call the undersigned attorney for any remaining issues.

S.N. 10/019,330
Art Unit: 2143

OIPE IAP79
APR 04 2007
PATENT & TRADEMARK OFFICE

Respectfully submitted:

[illegible]
Robert J. Mauri
Reg. No.: 41,180

3/30/07
Date

Customer No.: 29683

HARRINGTON & SMITH, PC
4 Research Drive
Shelton, CT 06484-6212

Telephone:	(203)925-9400
Facsimile:	(203)944-0245
email:	rmauri@hspatent.com

CERTIFICATE OF MAILING OR TRANSMISSION

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, P.O. BOX 1450, Alexandria, VA 22313-1450 or is being transmitted electronically to the United States Patent and Trademark Office.

Ann Okrentowich
Name of Person Making Deposit

3-30-07
Date